CKO2 - XML Protocols

CKO2 outputs - A2A general description

Author:	NBB - IT Department
Service:	IT Applications - PRSM
Date:	12/05/2011
Document version:	V0.4

BanqueNationale
DE BELGIQUE
Eurosystème

Table of Contents

1. Introduction

This document provides a general description of the different services available to the participant to retrieve the CKO2 outputs.

1.1 Document history

Date	Version	Description of change
15/09/2010	0.1	draft
07/12/2010	0.2	update based on the new functionalities
25/02/2011	0.3	update based on the new services' XSDs and new URLs
12/05/2011	0.4	update based on the last modifications in XSDs and protocol

1.2 References

Ref.	Title
[01]	OneGate(CSSR) – Manuel d'utilisation pour le déclarant FR – version 11-2009.pdf
[02]	OneGate(CSSR) – Handleiding voor de aangever NL – versie 11-2009.pdf
[03]	Technical documentation – Participant's directory function – Description of the data returned – v1.0 – 01/08/2010.pdf
[04]	Technical documentation – Automatic return function – Description of the data returned – v1.0 – 01/08/2010.pdf
[05]	OneGate(CSSR) – Webservices – End User Manual – v2.0 – 10/09/2010.pdf
[06]	OneGate(CSSR) – HTTPS entrypoints – End User Manual – v2.0 – 10/09/2010.pdf
[07]	URLs Entrypoints CKO2.xls

1.3 Context

The goal of this document is to describe the different services proposed to the participants to retrieve the automatic return and participant directory via A2A. Participants searching for a detailed description of these outputs' content should consult references [03] and [04].

For U2A participants, a specific interface is being developed to provide an interactive view of these outputs and its attachment(s), however this html view works only for unzipped attachments. It is also possible, for U2A users, to get the attachment(s) in an XML format with a structure identical to the one proposed to A2A users. A detailed description of the U2A consultation of messages (including those containing outputs) is available in references [01] and [02].

2. CKO2 outputs definition

This chapter describes in generic terms the two outputs that are already fully defined: automatic return and participant's directory. Sectorial and geographical, selective or historical statistics can be supplied to participants on request or periodically. Since those outputs are yet to be defined (content, format, frequency …) and since they will be delivered only via U2A, they are not part of this document.

Other messages will also be available (e.g. CKO2 back-office communication messages) but their structure won't be necessarely defined. It will be possible to retrieve those messages via A2A using the services described in this document. However, it will be very difficult to process them automatically since their structure will be unknown. This is why each A2A participant should have at least one U2A certificate in order to be able to read those messages via the web interface. A2A participants will be able to differentiate the different type of messages by checking the "body" (see chapter 4.5.3 for more details).

2.1 Automatic return

Definition

The automatic return contains the credit situation of the participant's debtors at the participant side and for all the participants based on the two most recent complete[1] reporting months as far as the debtors have credits for the last reporting month. It could be that no data is present in the output (but only a specific information message) for one or two months, although the participant subscribed to it: the reason is that the participant did not report any data for that month.

Frequency

The automatic return is a monthly output which is delivered to a participant if he has subscribed to it. This subscription must be done via a specific form to be filled in OneGate.

Format

The automatic return default format is an XML file. However, if the XML file contains few debtors (up to maximum 100 debtors) and if the participant has requested it when filling the request form, the automatic return will also be delivered in an HTML format (in French, Dutch or both). The participant can get the automatic return in the attachment(s) of a message with a specific body (message subject) in his incoming mailbox. The automatic return file(s) in XML format will always be zipped (see chapter 3 for more details concerning volumes and compression). If the participant has requested HTML format, the HTML files won't be zipped.

[1] The "completeness" of a reporting month is evaluated by the CKO2 back-office based on the number of reporting participants who have already reported the expected amount of data.

2.2 Participant’s directory

Definition

The participant’s directory contains all the data reported by the participant for one month chosen by the participant among the last 12 available months.

Frequency

The participant’s directory is a unique output which can be obtained on demand via a specific form to be filled in OneGate. If the participant wants to request several months of data, he must fill in a specific form for each month.

Format

The participant’s directory default format is an XML file. However, if the XML file contains few debtors (up to maximum 100 debtors) and if the participant has requested it when filling the request form, the participant’s directory will also be delivered in an HTML format (in French, Dutch or both). The participant can get the participant’s directory of a specific month in the attachment(s) of a message with a specific body (message subject) in his incoming mailbox. The participant’s directory file(s) in XML format will always be zipped (see chapter 3 for more details concerning volumes and compression). If the participant has requested HTML format, the HTML files won’t be zipped[1].

[1] This solution was implemented to guarantee an HTML view of these attachments via the interactive web application.

3. Volumes and compression

Volumes and compression problems concern only xml attachments. If the output contains few debtors and the participant has requested it, the output will also contain one or several html attachments that won't be zipped. Therefore, rules defined here below are applied only to xml attachments.

As defined in the references [03] and [04], automatic return and participant's directory xml attachments will always be zipped.

Some participants will send large volumes of report data and will then receive large automatic return and participant's directory files. For them compression (zipping) will not be enough and splitting of the output files will be necessary to avoid problems related to sending and processing large files (long transfer times, timeouts, server memory problems, …).

Output xml attachments will therefore always be zipped and, "only when necessary" (= to avoid problems), split in several smaller files, as described by the following rules:

1. The compression algorithm used will be the popular "zip", which corresponds to the content type "application/zip"[1]. So participants can unzip using
   - WinZip (even older versions), always in "legacy mode", no encryption, no extra features.
   - Zipx will probably not be able to unzip the output files.
   - Tests with gZip proofed it is also capable of doing the unzip[2].
2. An "output file size splitting limit" parameter will be used to determine if an output file is too big (before compression) to send without splitting it into several smaller files. If the size before compression of an output file to be sent is greater than this parameter, splitting will be performed. The initial value of this parameter will be 1Mb and remain so until a new value will be communicated to all participants. 1Mb being the maximum, most files will have a size lower than 900Kb unzipped.
3. The participants will be able to detect if splitting has been performed by using the output sequence number defined in the administration part of the output XML file. An XML file with the total output sequence number equal to "1" has not been split in several smaller files.
4. The output files will not be present in the replies of the webservices or HTTPS entrypoints used to request outputs, only URL's referring to the output files will be present. With each URL the participant can then fetch the corresponding output file via simple HTTPS GET request.
5. After unzipping an attachment, the type of the content of the unzipped attachment will be revealed by the file extension: "xml".

Example: following the defined rules, a participant with an output file of 10Mb (unzipped) will receive 10 or more files of maximum 1Mb (unzipped).

[1] http://www.iana.org/assignments/media-types/application/zip

[2] Because the underlying compression algorithm is "deflate", used in WinZip and gZip.

4. Generic output services

This chapter describes the different webservices and HTTPS entrypoints provided to get an output file. The webservices use the SOAP protocol.

4.1 Introduction

To fetch an output file, the data exchange with the OneGate application consists of three chronological activities:

- The participant requests the list of available messages of a defined type[1].
- The participant requests a specific message (for each of the available messages).
- The participant submits an HTTPS request with each of the URL's of the different attachments in the specific message and the server responds with the content of the attachment.

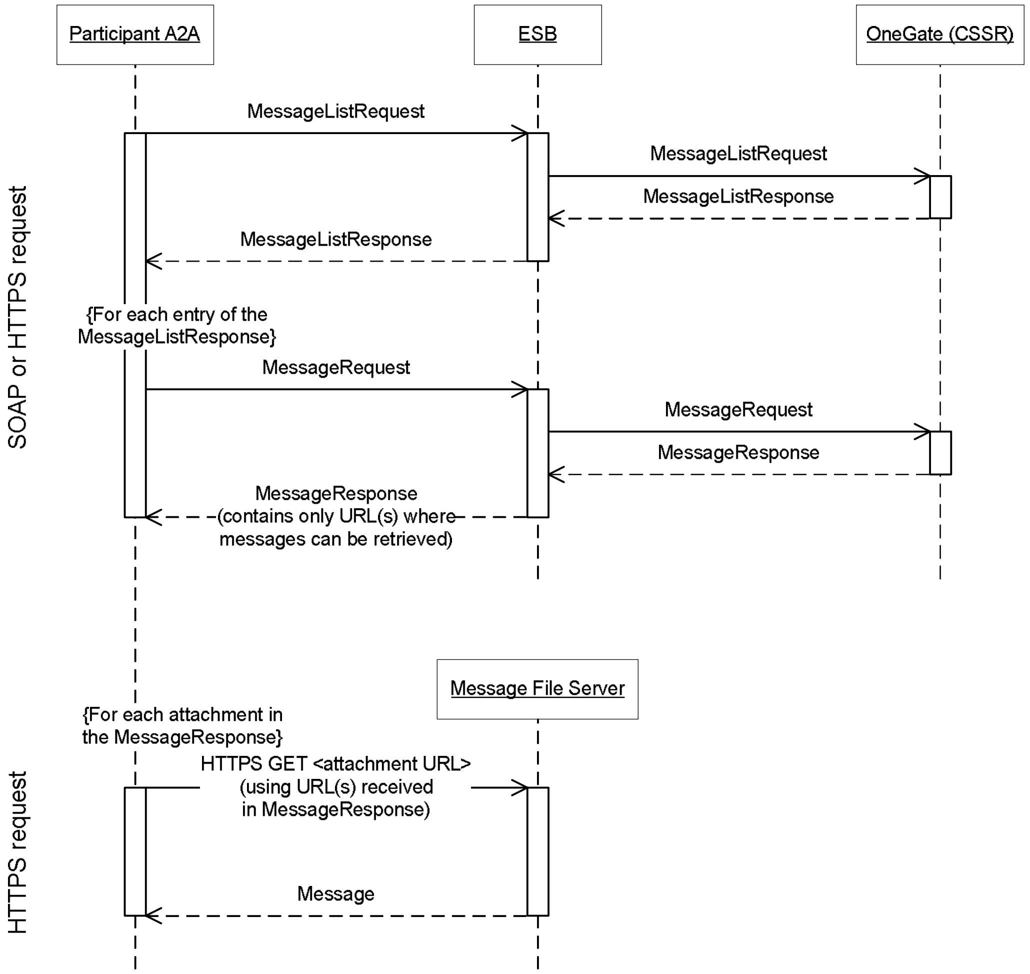

Participant A2A
ESB
OneGate (CSSR)
MessageListRequest
MessageListRequest
MessageListResponse
MessageListResponse
SOAP or HTTPS request
{For each entry of the MessageListResponse}
MessageRequest
MessageRequest
MessageResponse
MessageResponse (contains only URL(s) where messages can be retrieved)
Message File Server
{For each attachment in the MessageResponse}
HTTPS GET <attachment URL> (using URL(s) received in MessageResponse)
HTTPS request
Message

Figure 1 – Data exchange with OneGate for A2A message retrieving process

[1] "Message type" will be defined in a later chapter of this document.

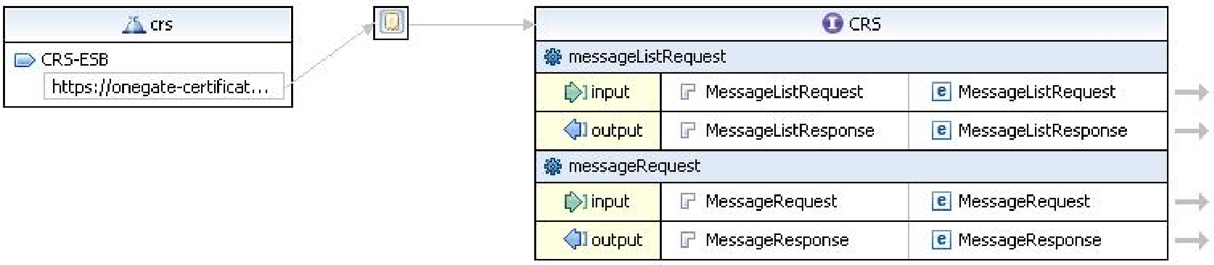
The content of the MessageResponse is one (or several) URL(s) where the message (output file) can be retrieved. The message can be retrieved using an HTTPS GET request to this URL. It implies that, if the participant works with SOAP requests for the two first steps, he must use two different sessions, SOAP and HTTPS sessions, and must authenticate in both sessions.

More detailed information concerning the webservices can be found in reference [05] while more detailed information concerning the HTTPS entrypoints is available in reference [06].

The file retrieved with the HTTPS GET request will have the structure defined in reference [03] and [04].

A message is flagged as “read” in the CKO2 application when the participant has successfully performed the second step (= request a specific message). The “read” status is stored for a specific NUIN meaning that, if a participant has retrieved a specific message with his A2A certificate, the message will appear as “not read” when retrieving it via a U2A certificate.

4.2 Webservices overview



crs

CRS-ESB

https://onegate-certificat...

CRS		
messageListRequest		
input	MessageListRequest	MessageListRequest
output	MessageListResponse	MessageListResponse
messageRequest		
input	MessageRequest	MessageRequest
output	MessageResponse	MessageResponse

Figure 2 – Webservices described in the WSDL

Figure 2 gives an overview of the webservices used by the participant to fully automate the message retrieving process data exchange with the OneGate application.

4.3 HTTPS entrypoints overview

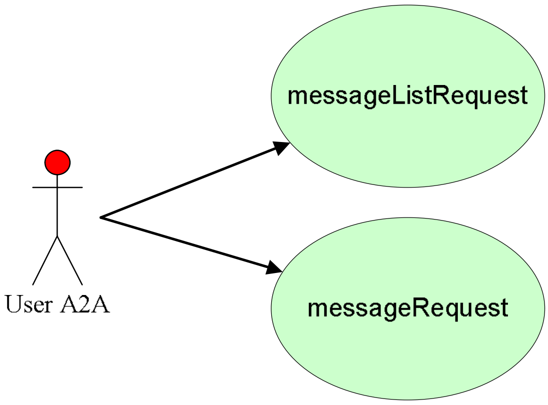


Figure 3 – HTTPS entrypoints services

Figure 3 gives an overview of the HTTPS entrypoints used by the participant to fully automate the message retrieving process data exchange with the OneGate application.

4.4 Request list of available messages (SOAP and HTTPS requests)

4.4.1 Description

The service “messageListRequest” is used to request the list of message identifiers available. Only the identifiers of messages destined to this userId will be sent back. Messages associated with the userId (certificate) of another user but for a common declarer will not be sent back. It is therefore recommended to participants to use the same A2A certificate on each of their servers to simplify the messages retrieval process.

The participant can choose between requesting a list of either new messages (not read by the requesting user) or messages already read/retrieved during a specified time frame. The second option offers the possibility to request messages that have been retrieved earlier. Messages are available during a period of 3 months.

The participant must define the type of message he wants to retrieve. The type of message can be one of the following values:

- ALL – messages of all types
- APP – messages intended for everyone who uses the application
- DOM – messages intended for all users who have access to the specified domain
- RPT – messages intended for all users who can report for a specific report
- DCL – messages intended for all users who can report for a specified declarer
- DCD – messages intended for all users who can report for a specified declarer whitin a specified domain
- FDT – messages intended for a user who has reported for a specific form and for a specific date (period)
- UID – messages intended for a specific user

Automatic return and participant directory are messages of type “DCD”.

The participant can call the “messageListRequest” service any time he wants[1], each time he will receive the list of identifiers of messages that fill the request parameters and are available for reading/download. If no message matching the request parameters is currently available for reading/download, the participant will receive a “NoMessage” answer.

The number of message identifiers returned by the “messageListRequest” service is limited to a maximum of 5000 message identifiers in order to limit the size of the answer to maximum 1Mb. Therefore, in order to receive all his message identifiers, a participant susceptible of having more than 5000 available messages should

- request the list of available messages
- request each available message received in the previous step
- start again steps 1 and 2 untill an empty list is received

Remark: the “messageListRequest” service will always return the 5000 first available messages meaning that, if a participant has 5001 available message, he has to read at least one available message by using the “requestMessage” service before being able to retrieve the 5001st message identifier in the list of available messages.

[1] Except during OneGate closing time, from 8h30 to 13h30 on Sundays.

4.4.2 Input

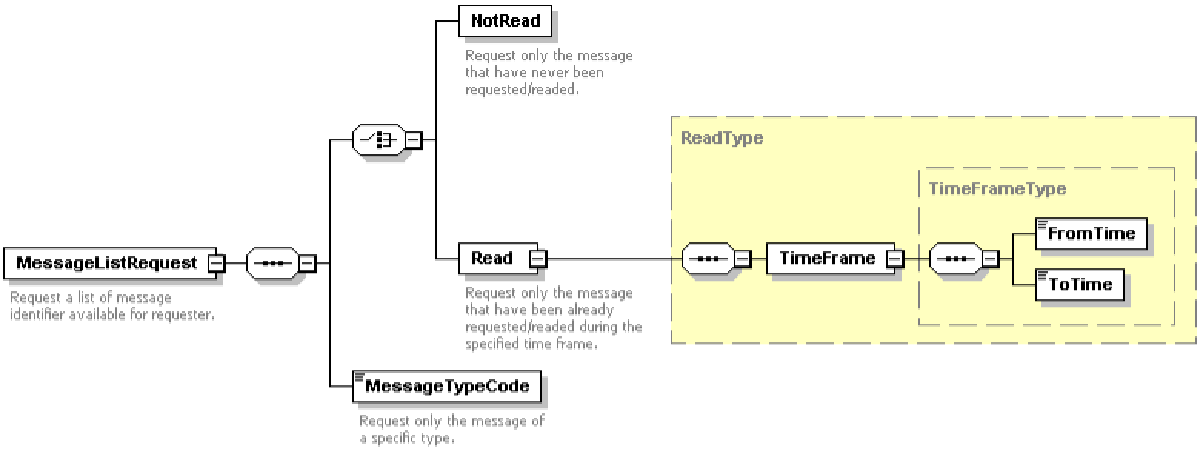

NotRead
Request only the message that have never been requested/readed.
ReadType
TimeFrameType
FromTime
ToTime
MessageListRequest
Request a list of message identifier available for requester.
Read
Request only the message that have been already requested/readed during the specified time frame.
TimeFrame
MessageTypeCode
Request only the message of a specific type.

Figure 4 – MessageListRequest

Element	Description
MessageListRequest	Contains the attributes to request the list of message identifiers available for the requester
NotRead	Used to request the identifiers of new messages (not read by requesting user)
Read	Used to request identifiers associated to messages already retrieved by requesting user earlier during the specified time frame
TimeFrame	Used to specify the time frame during which the messages have been retrieved the first time
FromTime	Contains the timestamp identifying the start of the time frame
ToTime	Contains the timestamp identifying the end of the time frame
MessageTypeCode	Used to specify the type of message requested

Table 1 – MessageListRequest elements description

4.4.3 Output

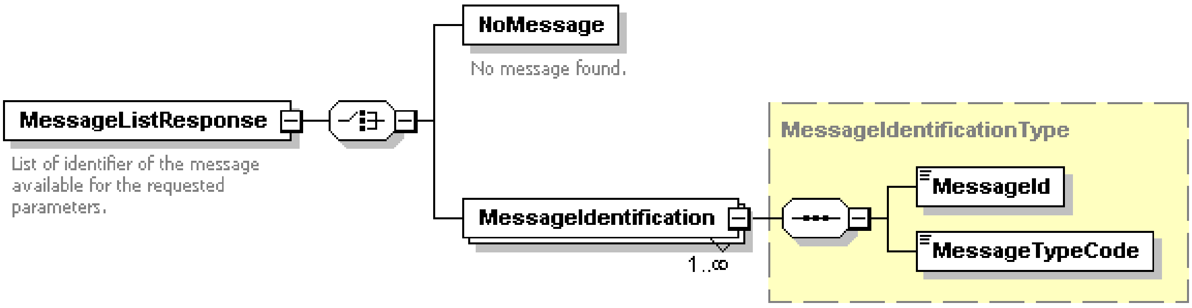

NoMessage
No message found.
MessageListResponse
List of identifier of the message available for the requested parameters.
MessageIdentificationType
MessageId
MessageIdentification
1..∞
MessageTypeCode

Figure 5 – MessageListResponse

Element	Description
MessageListResponse	Contains the list of available message identifiers
NoMessage	No message of the requested type found for the specified search criteria. • Not read: no new message • Read: no message read/retrieved during the specified time frame
MessageIdentification	Contains the identification data of the message
MessageId	Contains the identifier of the message (internal OneGate identifier)
MessageTypeCode	Contains the type of the message

Table 2 – MessageListResponse elements description

4.4.4 Error messages

4.4.4.1 SOAP error messages

Error message	Corrective action
Validation error	The SOAP request is not valid against the message definition. Please consult the validation error details.
Identification with code "userId" not found	You have access to OneGate but not for the requested data. • Check that you use the right URL[1] to access the webservices • Contact the CCCR back-office to request if you have access for the CCCR institute and your specific business domain (CC1)

Table 3 – MessageListRequest SOAP error messages description

4.4.4.2 HTTPS error messages

Status code – Reason phrase	Corrective action
400 Identification with code "userId" not found	You have access to OneGate but not to the requested institute. • Check that you use the right URL • Contact the CCCR back-office to request if you have access for the CCCR institute and your specific business domain (CC1)
4xx range[2]	Client error. The request contains bad syntax or cannot be fulfilled.
5xx range	Server error. The server failed to fulfill the request. Please try again later and if the problem persists, contact the CCCR back-office.

Table 4 – MessageListRequest HTTPS error messages description

4.5 Request message (SOAP and HTTPS requests)

4.5.1 Description

The service "messageRequest" is used to request a specific message by providing its identifier received in the reply of the "messageListRequest" service.

4.5.2 Input

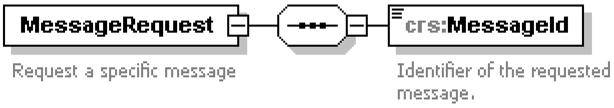

MessageRequest
Request a specific message
crs:MessageId
Identifier of the requested message.

Figure 6 – MessageRequest

Element	Description
MessageRequest	Contains the information about the requested message
MessageId	Contains the identifier of the requested message (internal OneGate identifier)

Table 5 – MessageRequest elements description

[1] In order to access OneGate in A2A, a specific URL must be used.
[2] An overview of HTTPS error codes can be found in http://www.rfc-editor.org/rfc/rfc2616.txt

4.5.3 Output

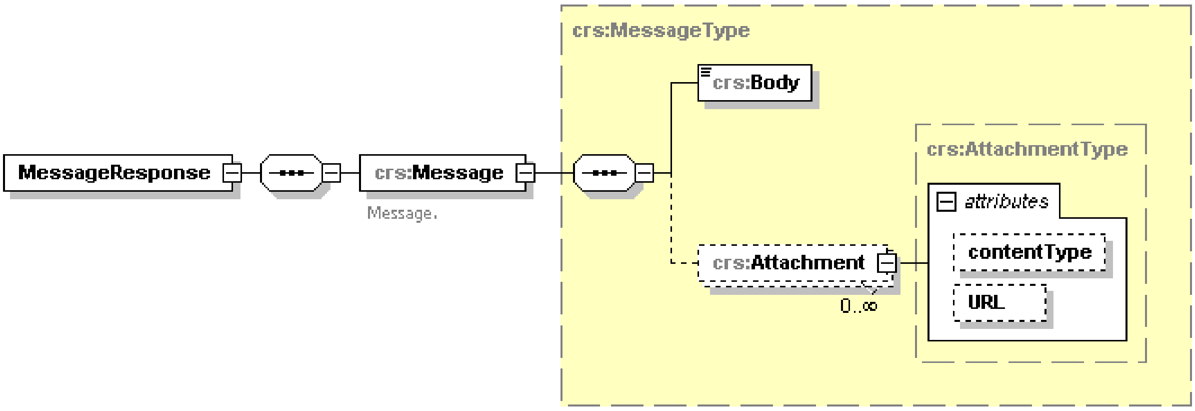


Figure 7 – MessageResponse

Element	Description
MessageResponse	Contains the requested message
Message	Contains at least one body, and can also contain one or several attachments
Body	Contains the body of the message in plain text which describes briefly the type and content of the message
Attachment	Contains an attachment of the message
@contentType	Specifies the type of the attachment's content using the Internet media type[1], it will always be "application/zip"
@URL	Contains the URL to which the message can be retrieved using an HTTPS GET request

Table 6 – MessageResponse elements description

[1] An Internet media type, originally called MIME type, is a two-part identifier for file formats on the internet. A media type is composed of two parts: a type and a subtype (e.g. text/plain, image/jpeg, application/pdf).

The body of the message will always have one of the two following structure:

- CKO2 output for {REPORT;aaaa} {PRODUCT_CODE;bbbb} {PERIOD;cccc} {FORMAT;dddd} {APPLICATION;eeee} {PARTICIPANT_CODE;hhhh} {OUTPUT_CREATION_TIMESTAMP;iiii}
- CKO2 output for {REPORT;aaaa} {PRODUCT_CODE;bbbb} {PERIOD1;cccc} {PERIOD2;dddd} {FORMAT;eeee} {APPLICATION;ffff} {PARTICIPANT_CODE;iiii} {OUTPUT_CREATION_TIMESTAMP;jjjj}

This structure regroups several information:

- REPORT: indicates the type of the output (e.g. "PARTICIPANT_DIRECTORY").
- PRODUCT_CODE: indicates the code of the product.
- PERIOD: indicates the period (format YYYY-MM) for which the output is generated.
- PERIOD1: indicates the first period (format YYYY-MM) for which the output is generated, it is only used when the output concerns two periods.
- PERIOD2: indicates the second period (format YYYY-MM) for which the output is generated, it is only used when the output concerns two periods.
- FORMAT: indicates the format of the output. It will always be "application/zip" for automatic return and participant directory. If several attachment have a different format (e.g. participant has requested the output in HTML in addition to the usual XML format), only the format of the main file (always "application/zip" for automatic return and participant's directory) is mentioned.
- APPLICATION: indicates the code of the application, will always be "CCCR".
- PARTICIPANT_CODE: indicates the code of the participant (e.g. "1500").
- OUTPUT_CREATION_TIMESTAMP: contains the timestamp (format YYYY-MM-DDThh:mm:ssZ) of the output creation.

Here follows two examples of message bodies:

- CKO2 output for {REPORT;AUTOMATIC_RETURN} {PRODUCT_CODE;0123} {PERIOD1;2010-05} {PERIOD2;2010-06} {FORMAT;application/zip} {APPLICATION;CCCR} {PARTICIPANT_CODE;0123} {OUTPUT_CREATION_TIMESTAMP;2010-12-01T10:15:35Z}
- CKO2 output for {REPORT;PARTICIPANT_DIRECTORY} {PRODUCT_CODE;0125} {PERIOD;2010-08} {FORMAT;application/zip} {APPLICATION;CCCR} {PARTICIPANT_CODE;1500} {OUTPUT_CREATION_TIMESTAMP;2010-12-01T10:15:35Z}

Remark: the combination of "PARTICIPANT_CODE" and "OUTPUT_CREATION_TIMESTAMP" elements constitutes the "CCCR_ID" element found in the administration part of the xmls and that is the unique identifier of the output.

4.5.4 Error messages

4.5.4.1 SOAP error messages

Error message	Corrective action
Validation error	The SOAP request is not valid against the message definition. Please consult the validation error details.
Identification with code “userId” not found	You have access to OneGate but not to the requested data. • Check that you use the right URL[1] to access the webservices • Contact the CCCR back-office to request if you have access for the CCCR institute and your specific business domain (CC1)

Table 7 – MessageRequest SOAP error messages description

4.5.4.2 HTTPS error messages

Status code – Reason phrase	Corrective action
400 Identification with code “userId” not found	You have access to OneGate but not for the requested institute. • Check that you use the right URL • Contact the CCCR back-office to request if you have access for the CCCR institute and your specific business domain (CC1)
400 Identification with code “messageId” not found	The requested message doesn’t exist.
4xx range	Client error. The request contains bad syntax or cannot be fulfilled.
5xx range	Server error. The server failed to fulfill the request. Please try again later and if the problem persists, contact the CCCR back-office.

Table 8 – MessageRequest HTTPS error messages description

[1] In order to access OneGate in A2A, a specific URL must be used.

4.6 Request attachment

4.6.1 Description

Each attachment of a MessageResponse should be retrieved by its unique URL using an HTTPS GET request.

4.6.2 HTTPS request

The attachment can be accessed automatically (in the client application) or semi-automatically (by using cURL for example) by performing a "GET" operation on the URL.

4.6.3 Error message

Status code – Reason phrase	Corrective action
4xx range	Client error. The request contains bad syntax or cannot be fulfilled.
5xx range	Server error. The server failed to fulfill the request. Please try again later and if the problem persists, contact the CCCR back-office.

Table 9 – Get attachment HTTPS error messages description

4.6.4 Remarks

The execution of the "Request attachment" requires an HTPPS session (a SOAP session is not appropriate). Therefore if SOAP is used for MessageListRequest and MessageRequest webservices utilization, the participant will have to deal with two types of sessions: SOAP and HTTPS. However, the same certificate can be used to give access for both session types.

5. Examples

5.1 SOAP

5.1.1 MessageListRequest

Here follows a typical SOAP request to retrieve identifiers of an already read message. In this example the participant would like to get all the identifiers of DCD messages already retrieved from 2010-06-09 to 2010-06-15. Dates must be defined using the full dateTime syntax (YYYY-MM-DDThh:mm:ss).

Messages list SOAP request:

```
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:esb="http://www.onegate.eu/2010-01-01/esb">
   <SOAP-ENV:Header />
   <SOAP-ENV:Body>
      <esb:MessageListRequest>
         <esb:Read>
            <esb:TimeFrame>
               <esb:FromTime>2010-06-09T00:00:00</esb:FromTime>
               <esb:ToTime>2010-06-15T23:59:59</esb:ToTime>
            </esb:TimeFrame>
         </esb:Read>
         <esb:MessageTypeCode>DCD</esb:MessageTypeCode>
      </esb:MessageListRequest>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The result would look like this:

Messages list SOAP response:

```
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:esb="http://www.onegate.eu/2010-01-01/esb">
   <SOAP-ENV:Header />
   <SOAP-ENV:Body>
      <esb:MessageListResponse>
         <esb:MessageIdentification>
            <esb:MessageId>142</esb:MessageId>
            <esb:MessageTypeCode>DCD</esb:MessageTypeCode>
         </esb:MessageIdentification>
         <esb:MessageIdentification>
            <esb:MessageId>148</esb:MessageId>
            <esb:MessageTypeCode>DCD</esb:MessageTypeCode>
         </esb:MessageIdentification>
         <esb:MessageIdentification>
            <esb:MessageId>195</esb:MessageId>
            <esb:MessageTypeCode>DCD</esb:MessageTypeCode>
         </esb:MessageIdentification>
      </esb:MessageListResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

A typical SOAP request to retrieve the identifiers of unread DCD messages would be similar to the following example.

Messages list SOAP request:

```
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:esb="http://www.onegate.eu/2010-01-01/esb">
   <SOAP-ENV:Header />
   <SOAP-ENV:Body>
      <esb:MessageListRequest>
         <esb:NotRead />
         <esb:MessageTypeCode>DCD</esb:MessageTypeCode>
      </esb:MessageListRequest>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

5.1.2 MessageRequest

Here follows a typical SOAP request to retrieve the message for a given identifier. The example retrieves the message associated with identifier 148:

Message SOAP request:

```
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:esb="http://www.onegate.eu/2010-01-01/esb">
   <SOAP-ENV:Header />
   <SOAP-ENV:Body>
      <esb:MessageRequest>
         <esb:MessageId>148</esb:MessageId>
      </esb:MessageRequest>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The result would look like this (the content of the body has been reduced to improve readability):

Message SOAP response with several attachments:

```
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:esb="http://www.onegate.eu/2010-01-01/esb">
   <SOAP-ENV:Header />
   <SOAP-ENV:Body>
      <esb:MessageResponse>
         <esb:Message>
            <esb:Body>CKO2 output for …</esb:Body>
            <esb:Attachment contentType="application/zip" URL="https://onegate-cccr-a2a-
test.nbb.be/crs/a2a/message/attachment/142254" />
            <esb:Attachment contentType="application/zip" URL="https://onegate-cccr-a2a-
test.nbb.be/crs/a2a/message/attachment/128564" />
         </esb:Message>
      </esb:MessageResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The URL(s) will refer to an URL that is accessible on the Websphere Application Server (WAS). This implies that an A2A authenticated user that has access to the webservices will also be given the ability to authenticate to the WAS application and get access to the requested URL. The application also verifies that the A2A user has the authority to access the specified URL. This is done to prevent a participant from accessing data of another participant.

The attachment URL can be accessed automatically (in the client application) or semi-automatically (by using cURL or SoapUI) by performing an HTTPS GET request on the URL.

5.2 cURL

cURL is a common free software that allows participants to communicate with the CKO2 application using the HTTPS entrypoints.

5.2.1 Parameters

--data-binary <data>

(HTTP) This posts data exactly as specified with no extra processing whatsoever. If you start the data with the letter @, the rest should be a filename. Data is posted in a similar manner as --data-ascii does, except that newlines are preserved and conversions are never done.

-E/--cert <certificate[:password]>

(SSL) Tells cURL to use the specified certificate file when getting a file with HTTPS or FTPS. The certificate must be in PEM format. If the optional password isn't specified, it will be queried for on the terminal. Note that this option assumes a "certificate" file that is the private key and the private certificate concatenated! See --cert and --key to specify them independently.

-H/--header <header>

(HTTP) Extra header to use when getting a web page. You may specify any number of extra headers. Note that if you should add a custom header that has the same name as one of the internal ones cURL would use, you externally set header will be used instead of the internal one.

-v/--verbose

Makes the fetching more verbose/talkative. Mostly useful for debugging.

5.2.2 Request list of messages

Command cURL:

```
curl –v –E "<certificate.pem:password>" --data-binary "@requestMessageList.xml"
-k "https://onegate-cccr-a2a-test.nbb.be/cccr-esb/invoke/requestMessageList-v2-0"
```

HTTPS request (requestMessageList.xml) example:

```
<?xml version="1.0"?>
<esb:MessageListRequest xmlns:esb="http://www.onegate.eu/2010-01-01/esb">
   <esb:NotRead />
   <esb:MessageTypeCode>DCD</esb:MessageTypeCode>
</esb:MessageListRequest>
```

HTTPS response example:

```
<?xml version="1.0"?>
<esb:MessageListResponse xmlns:esb="http://www.onegate.eu/2010-01-01/esb">
   <esb:MessageIdentification>
      <esb:MessageId>125</esb:MessageId>
      <esb:MessageTypeCode>DCD</esb:MessageTypeCode>
   </esb:MessageIdentification>
   <esb:MessageIdentification>
      <esb:MessageId>154</esb:MessageId>
      <esb:MessageTypeCode>DCD</esb:MessageTypeCode>
   </esb:MessageIdentification>
</esb:MessageListResponse>
```

5.2.3 Request message

Command cURL:

```
curl –v –E “<certificate.pem:password>” --data-binary “@requestMessage.xml”
-k “https://onegate-cccr-a2a-test.nbb.be/cccr-esb/invoke/requestMessage-v2-0”
```

HTTPS request (requestMessage.xml) example:

```
<?xml version="1.0"?>
<esb:MessageRequest xmlns:esb="http://www.onegate.eu/2010-01-01/esb">
   <esb:MessageId>154</esb:MessageId>
</esb:MessageRequest>
```

HTTPS response example (the content of the body has been reduced to improve readability):

```
<?xml version="1.0"?>
<esb:MessageResponse xmlns:esb="http://www.onegate.eu/2010-01-01/esb">
   <esb:Message>
      <esb:Body>CKO2 output for …</esb:Body>
      <esb:Attachment contentType="application/zip" URL="https://onegate-cccr-a2a-
test.nbb.be/crs/a2a/message/attachment/142254" />
      <esb:Attachment contentType="application/zip" URL="https://onegate-cccr-a2a-
test.nbb.be/crs/a2a/message/attachment/128564" />
   </esb:Message>
</esb:MessageListResponse>
```

5.2.4 Get attachment for message

Command cURL:

```
curl –v –E “<certificate.pem:password>”
-o “https://onegate-cccr-a2a-test.nbb.be/crs/message/attachment/142254”
```

The HTTPS response is a zipped XML file with the structure defined in reference [03] for participant’s directory or [04] for automatic return or an HTML file if the participant has requested it.